



**CONGESTED HIGHWAYS ACTION RESPONSE TEAM  
STATE HIGHWAY ADMINISTRATION**

---

## **CHART II Java Feasibility Investigation**

**Contract DBM-9713-NMS  
TSR # 9901961  
Document # M361-AR-003R0**

**July 1, 1999  
By  
Computer Sciences Corporation and PB Farradyne Inc**



# Table of Contents

---

<b>1</b>	<b>Introduction .....</b>	<b>3</b>
<b>2</b>	<b>Results of Investigation by Task.....</b>	<b>4</b>
<b>2.1</b>	<b>Map User Interface .....</b>	<b>4</b>
2.1.1	In-House Development .....	4
2.1.2	Third Party Tools .....	5
2.1.2.1	ESRI Internet Map Server.....	5
2.1.2.2	Autodesk MapGuide .....	5
2.1.2.3	PlexStar ClearSpace.....	6
<b>2.2</b>	<b>Add Drag/Drop Capabilities To Existing Java/CORBA GUI .....</b>	<b>6</b>
<b>2.3</b>	<b>Set Up Internal Web Server .....</b>	<b>6</b>
<b>2.4</b>	<b>Determine Best Method Of GUI Implementation And Software Distribution .....</b>	<b>6</b>
<b>2.5</b>	<b>Perform Proof Of Concept Using Existing Java CORBA GUI Hosted On Internal Web Server .....</b>	<b>7</b>
<b>2.6</b>	<b>Investigate Available Text-To-Speech-To-Text Products For Java.....</b>	<b>7</b>
<b>2.7</b>	<b>Approach Document For Text-To-Speech GUI.....</b>	<b>7</b>
<b>2.8</b>	<b>Document Describing Web User Interface Approach .....</b>	<b>7</b>
<b>3</b>	<b>Conclusions.....</b>	<b>8</b>

# 1 Introduction

The purpose of this document is to provide the results or task disposition for each item of the web feasibility investigation for the CHART II project. The document discusses each item from the work breakdown structure for the investigation and provides a summary of actions performed and/or conclusions made by the development team. The work breakdown structure is attached below.

ID		Task Name	Duration
1		<b>Web Based GUI</b>	<b>70 days</b>
2		<b>Map</b>	<b>38 days</b>
3		<b>In House Development</b>	<b>38 days</b>
4		Verify Primitive and Bitmap support	2 wks
5		Random Line Test	3 wks
6		Miscellaneous view topics	1 wk
7		Frame Test	1 wk
8		Drawing on separate thread	3 days
9		<b>Third Party Tool</b>	<b>35 days</b>
10		Research and Evaluate Vendor Products/APIs	2 wks
11		Select and acquire a vendor product for proof of concept	2 wks
12		Perform Map proof of concept with selected vendor product	2 wks
13		Verify integration with GUI	1 wk
14		Add drag and drop capabilities to existing Java/CORBA GUI	2 days
15		Setup internal web server	3 days
16		Determine Best Method of Web based implementation/distribution	3 wks
17		Perform proof of concept using existing Java/CORBA GUI hosted on internal web server	2 wks
18		Investigate available Text-to-speech-to-text products for Java	2 wks
19		Approach document for text-to-speech GUI	2 days
20		Document describing intended Web user interface approach	1 wk

## **2 Results of Investigation by Task**

---

The investigation was initiated because the development team, during analysis of CHART project requirements identified issues that indicated that the use of the Java programming environment rather than the C++ environment which was initially proposed, may be more beneficial to the CHART II software development project. This is because of Java's ease of use for developers and its multi-platform capabilities. However the team also identified critical software functions that might be difficult or impossible to implement efficiently using Java. Foremost among these was the graphical map user interface. The investigation was thus targeted at resolving what were identified as high-risk tasks for Java programming. During the course of the investigation and associated project efforts, all of the identified high-risk tasks were tested using prototype development or were deemed to no longer be high risk based upon information discovered by the development team.

### **2.1 Map User Interface**

Initially, it was anticipated that there were two viable approaches for providing the Map User Interface capability, in-house development based upon adaptation of previous PBFI map interface software and procurement of off-the-shelf Map User Interface software. Results are provided below.

#### **2.1.1 In-House Development**

In order to test the feasibility of converting PBFI's Map User Interface to the Java environment, sample map applications were written using Java and the following graphics library packages: Java 2D, Java 3D, GL4Java (OpenGL wrapper), Magician (OpenGL wrapper). Library packages provide software functions that make it easier for the programmer to draw and manipulate graphics on the computer screen without having to write program code for specific hardware. One standard graphics library is OpenGL. Other libraries wrap the OpenGL library functions in higher level functions that remove the software even further from dependence on specific hardware and operating systems. Each of the aforementioned packages was tested for completeness and performance. The team determined from initial testing that Magician was the best candidate for continued feasibility testing due to its excellent performance and its complete support of the OpenGL API.

Performance of the Map User Interface is one of the critical requirements for CHART II. A random line test was performed to verify that the performance would be acceptable with a large number of small polylines to simulate what would be experienced when drawing a typical ITS map file. The results of the random line test indicated that neither use of the Java environment nor the library functions injected significant delays over what would be experienced with the same function using C++. The polyline performance test was easily passed.

Additional tests were conducted to verify that user interface functions such as tool tips and object selection could be implemented. After finding that the view topics would not pose a problem, the team tested Java frames to verify that they would not present a problem. The Java frame classes were capable of performing all necessary tasks to implement the map as it was demonstrated in the prototype. The team then implemented a separate draw thread in order to simulate a realistic map design. Again the test was successful, in fact the team found it easier to implement the map in Java than in C++.

### **2.1.2 Third Party Tools**

Several of third party tools were evaluated. This section briefly describes 3 viable candidates and lists the results of the investigation.

#### **2.1.2.1 ESRI Internet Map Server**

This is a product from one of the major GIS software vendors. It is a new part of their product line. It uses a client/server architecture meaning that the client sends coordinates and layer information to the server and the server returns a GIF (graphics file) for the client to render. This architecture cannot be guaranteed to render a map within the required 3-second period because the server could become a performance bottleneck. For this reason the product was eliminated from consideration.

#### **2.1.2.2 Autodesk MapGuide**

Autodesk is a producer of popular computer-aided design graphics software. The MapGuide product also uses client/server architecture as described above. Thus for the same reason the product was eliminated from consideration.

### **2.1.2.3 PlexStar ClearSpace**

This product was the only Java GIS toolkit identified as being publicly available in April of 1999 when the investigation was conducted. It has an excellent application-programming interface (API) which is well documented and very flexible. A copy of the product was acquired by downloading from the Internet for evaluation. The product API fulfilled its promise of flexibility and allowed easy creation of a test application which could display a DXF (standard format map file) file. However, the product has been created using the Java 2D graphics library that is included with the Java language. This architecture proved to be too slow to meet the CHART II map performance requirements. The product was eliminated from consideration.

## **2.2 Add Drag/Drop Capabilities To Existing Java/CORBA GUI**

Drag and drop capability is one of the expected features of the CHART II user interface as demonstrated by the GUI prototype. Initially there were concerns regarding whether this capability could be readily implemented using Java. During the course of its investigation work and using Java, the development team has encountered extensive documentation which indicates that drag and drop is available in the Java language as a standard function so there is greater confidence that it will function as required to meet project requirements. Therefore, it was determined that this work item is not necessary.

## **2.3 Set Up Internal Web Server**

This work item was added to support the following work items for GUI implementation and software distribution. It was team determined that the following items are not required at this time; therefore, the web server setup is currently not needed.

## **2.4 Determine Best Method Of GUI Implementation And Software Distribution**

This item involved two components. The first was to determine if the CHART II GUI should be implemented as a Java Applet or a Java Application. After researching problems related to CORBA and Java applets, the team determined that the CHART II GUI should be implemented as a Java application. This determination was made because applets suffer from virtual machine incompatibilities across browsers. Secondly, the CHART II project has already done substantial work to define the look and feel of the GUI. This look and feel can be preserved in an application approach but not in an applet approach.

The second component involved determining the best method for distributing client software updates. The team concluded that this effort is no different whether done for C++ or Java and the use of Java introduces no additional or unique complexities. Thus, this work will be done as part of the normal system development cycle and does not require a separate task as part of the Java feasibility study.

## **2.5 Perform Proof Of Concept Using Existing Java CORBA GUI Hosted On Internal Web Server**

This work item was intended to implement the results of the GUI Implementation and Software Distribution task for testing. Given that the GUI will be a Java application rather than applet, the only remaining concept to prove is the method for software distribution, which will be determined during the system development cycle.

## **2.6 Investigate Available Text-To-Speech-To-Text Products For Java**

When the Java investigation began there was a question regarding whether to be compatible, a Java version of a text-to-speech application is needed to satisfy the text-to-speech requirement in the system. During the course of the investigation it was determined that the language neutrality of CORBA makes it both possible and no more difficult to implement the text-to-speech function in C++ if need be. Thus, it was determined that a separate feasibility study for this item is not necessary. Any effort required for identifying available text-to-speech products will be done in the normal course of design and independent of the programming language.

## **2.7 Approach Document For Text-To-Speech GUI**

The purpose of this document was to capture the findings of the text-to-speech for Java investigation. Because this investigation is not necessary, the document is not required.

## **2.8 Document Describing Web User Interface Approach**

The purpose of this document was to describe the approach being taken for the Java user interface. This would include if the GUI would be an application or applet, how it would render map files, how it would perform text-to-speech, and how it would be updated when new releases are made available. A document of this nature can be made available if so desired. However, it has not been produced as a result of this effort because the approach being taken is not substantially different than the C++ approach that was initially intended. The team is utilizing Java in order to provide a cross-platform GUI and to capitalize on its superiority as a programming language particularly in a distributed application environment.

### **3 Conclusions**

---

The results of the investigation have indicated that risks for development of critical graphical user interface components for CHART II are not increased by using the Java development environment. Further it was determined that the mission-critical graphical map user interface can be developed using Java in a way that will meet requirements.